

# CT Praktikum: Interrupt Performance

## 1 Einleitung

In diesem Praktikum messen Sie die Latenz eines Interrupts und die Dauer einer ISR (Interrupt Service Routine). Die Latenz ist die Zeit vom Auslösen des Interrupts, d.h. dem Auftreten des Ereignisses, bis zum Start der dazugehörigen ISR.

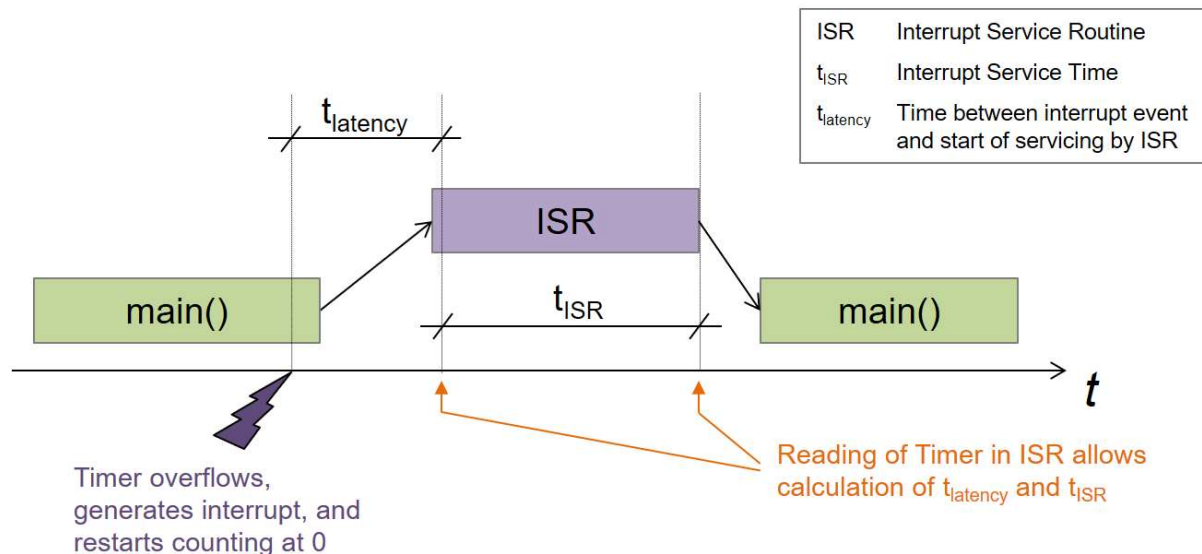


Abbildung 1: Messung der Latenz eines Interrupts mit einem Timer.

Die Latenz eines Interrupts lässt sich mit einem Timer messen. Der Timer ist als Upcounter konfiguriert und löst beim Erreichen des im Auto Reload Register (ARR) programmierten Wertes einen Interrupt aus. Der Timer beginnt danach direkt wieder bei null zu zählen. Wird das Zählregister direkt zu Beginn der Timer-ISR ausgelesen, kann damit die Latenz gemessen werden. Das Zählregister enthält dann die Anzahl Ticks, die seit dem Auslösen des Interrupts gezählt wurden. (Dabei wird hier vernachlässigt, dass nicht nur die Interrupt Latenz gemessen wird, sondern auch noch die Zeit, die benötigt wird um den Zähler auszulesen.)

## 2 Lernziele

- Sie können den Begriff der Interrupt-Latenz erklären und mögliche Ursachen nennen.
- Sie sind in der Lage, ein Programm zu realisieren um die Interrupt-Latenz zu messen.
- Sie verstehen wie andere, höher priorisierte Interrupts die Latenz eines Interrupts verlängern können.

### 3 Versuchsaufbau

Abbildung 2 zeigt den Versuchsaufbau.

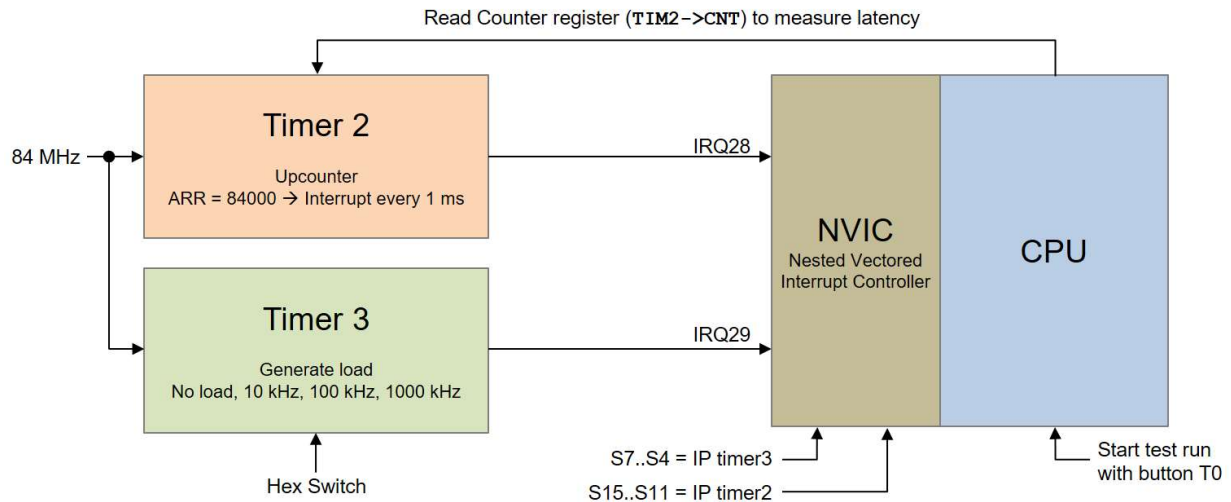


Abbildung 2: Versuchsaufbau

Es werden zwei Interrupt-Quellen verwendet:

- Quelle 1: Timer 2 ist als Upcounter konfiguriert und wird bei Erreichen des Reload-Wertes auf null zurückgesetzt. Er ist so konfiguriert, dass er alle 1 ms zurückgesetzt wird und einen periodischen Interrupt erzeugt.
- Quelle 2: Timer 3 dient zur Erzeugung einer Lastsituation. Mittels Hex Switch kann die Last eingestellt werden. Wird keine Last ausgewählt, so wird Timer 3 nicht gestartet.

Timer 2 und Timer 3 sind bereits konfiguriert. Sie finden die Konfigurationen im vorgegebenen Quelltext.

Gemessen wird in einer Endlos Schleife. Ein einzelner Testlauf wird durch Drücken der Taste T0 gestartet. Um Schwankungen zu eliminieren, mittelt jeder Testlauf die Daten über viele Timer 2 Interrupts. Im Endausbau soll unser System dann wie in Abbildung 3 gezeigt aussehen.

**Machen Sie sich mit der Struktur und dem Aufbau des gegebenen Codes vertraut.**

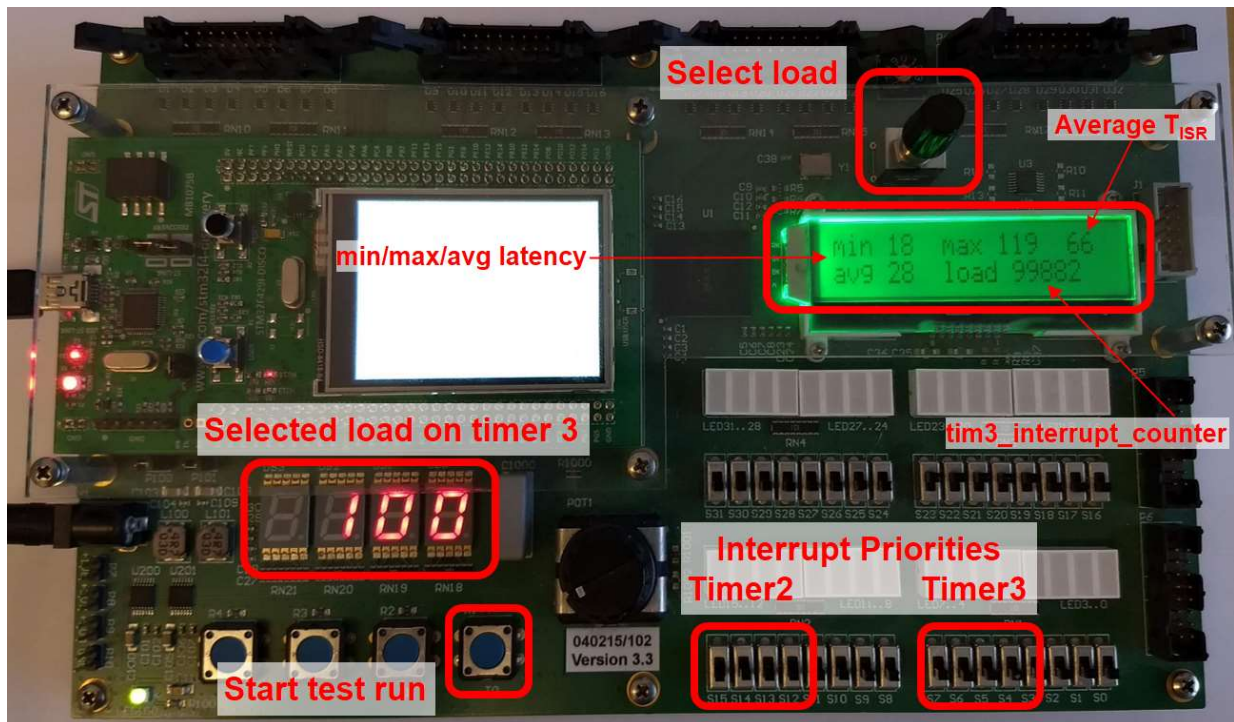


Abbildung 3: Endausbau auf CT-Board

## 4 Aufgaben

### 4.1 Interruptroutine zur Messung der Latenz

Zunächst sollen Sie die Latenz des Timer 2 Interrupts ohne Last messen. Timer 2 erzeugt bei Überlauf einen Interrupt. Dazu soll **zu Beginn** der entsprechenden ISR der Zählerstand des Timer 2 ausgelesen werden.

a) Ergänzen Sie die gegebene ISR für den Timers 2 wie folgt:

- Gleich am Anfang der ISR soll der Zählerstand des Timer 2 ausgelesen werden.
- Setzen Sie die IRQ Bedingung des Timers 2 zurück  
`hal_timer_irq_clear(TIM2, HAL_TIMER_IRQ_UE);`
- Inkrementieren Sie den vorgegebenen Zähler für die Anzahl Timer2 Interrupts
- Verwenden Sie den zuvor gespeicherten Zählerstand um die vordefinierten Variablen zu aktualisieren  
`min_latency` → Minimal aufgetretener Wert der Latenz,  
`max_latency` → Maximal aufgetretener Wert der Latenz und  
`sum_latency`. → Summe der Latenzen aus allen Interrupts. Dient am Ende des Testlaufs zur Berechnung des Durchschnittswerts.
- Sobald die definierte Anzahl Durchläufe erreicht ist (vordefinierte Konstante `NUMBER_OF_TIMER_2_INTERRUPTS`) stoppen Sie beide Timer (`hal_timer_stop()`) und setzen die zugehörige Boolesche Variable auf `TRUE`.

b) Testen Sie die Funktionalität Ihres Codes mit dem Debugger. Stellen Sie dazu den Hex Switch so ein, dass keine Last generiert wird, d.h. die Siebensegmentanzeige ist dunkel. Setzen Sie im Hauptprogramm einen Breakpoint beim Aufruf der noch zu schreibenden Funktion `print_results()`. Starten Sie den Test durch Drücken der Taste T0. Wie groß ist die Latenz minimal, maximal und im Durchschnitt? Tragen Sie

die Werte aus den entsprechenden Variablen in der unten stehenden Tabelle ein.

	Minimum	Maximum	Durchschnitt
Latenz (ticks)	21	32	21

## 4.2 Ausgabe der Werte auf dem Display

Geben Sie die Werte für Durchschnitt, Minimum und Maximum und die zugehörigen Beschriftungen auf dem Display des CT-Boards aus. Ergänzen Sie dazu die vorbereitete Funktion `print_results()`. Achten Sie auf die Kommentare in der Funktion.

## 4.3 Messung der Latenz unter Last

Nun wird Quelle 2 (Timer 3) verwendet um eine zusätzliche Last zu erzeugen. Dazu erzeugt der Timer 3 ebenfalls einen Interrupt, der je nach Einstellung des Hex Switches mit einer bestimmten Frequenz ausgelöst wird. Die dazugehörige ISR ist bereits implementiert. Darin wird lediglich ein Counter erhöht und etwas Verzögerung erzeugt.

- Ergänzen Sie Ihre Ausgabefunktion `print_results()` mit der Ausgabe der Anzahl Timer 3 Interrupts.
- Ergänzen Sie Ihren Code, um die Dauer der Interrupt Service Routine ( $T_{ISR}$ ) des Timers 2 zu messen und auf das LCD auszugeben. Lesen Sie dazu den Zählwert des Timers 2 am Ende der ISR nochmals aus. Die Differenz zum ersten Auslesen ergibt die gesuchte Dauer. Verwenden Sie die vordefinierte Variable `sum_tisr`, um die Werte für die Durchschnittsberechnung aufzusummieren.

Timer 2: Clock Source 84 MHz, prescaler = 1 (Register = 0)  
 -> Latenzzeit: 1 Tick =  $1 / (84 \text{ MHz}) = (1 / 84) \text{ us} = 11.9 \text{ ns}$

- c) Führen Sie die Messungen der Timer 2 Latenz mit verschiedenen Lastsituationen durch, d.h. für verschiedene Interrupt-Frequenzen des Timers 3. Tragen Sie die Werte (**Anzahl Ticks und Zeit**) in der untenstehenden Tabelle ein. Ermitteln Sie dazu den Takt des Timers 2 aus dem gegebenen Quelltext und **berechnen Sie die Latenzzeit**.

Interrupt-Frequenz Timer 3	Timer 2 Interrupt Priority Timer 2 = Interrupt Priority Timer 3			
	Minimum Latenz	Maximum Latenz	Durchschnitt Latenz	Durchschnitt $T_{ISR}$
	Anzahl ticks / Zeit in ns	Anzahl ticks / Zeit in ns	Anzahl ticks / Zeit in ns	Anzahl ticks / Zeit in ns
Timer 3 off Siehe 4.1b)	21	32	21	86
10 kHz	17	130	22	86
100 kHz	17	133	29	86
1 MHz	17	129	72	86

- d) Erklären Sie, warum die durchschnittliche Latenz des Interrupts für Timer 2 mit zunehmender Frequenz des Timer 3 Interrupts zunimmt.

Weil beide Timer den gleichen Interrupt-Controller beanspruchen und bei einer höheren Frequenz von Timer 3 häufiger Interrupts ausgelöst werden, welche abgearbeitet werden müssen

#### 4.4 Anpassung der Interrupt Prioritäten

Implementieren Sie das Einlesen und setzen der Interrupt Prioritäten am angegebenen Ort im Hauptprogramm.

Führen Sie die Messungen in den untenstehenden Tabellen durch.

<b>Timer 2</b> <b>Interrupt Priority Timer 2 &gt; Interrupt Priority Timer 3</b> PL(Timer 2) < PL(Timer 3)				
Interrupt-Frequenz Timer 3	Minimum Latenz <small>Anzahl ticks</small>	Maximum Latenz <small>Anzahl ticks</small>	Durchschnitt Latenz <small>Anzahl ticks</small>	Durchschnitt T <sub>ISR</sub> <small>Anzahl ticks</small>
Timer 3 off Siehe 4.1b)	21	32	21	86
10 kHz	17	30	21	86
100 kHz	17	30	21	86
1 MHz	16	30	21	86

<b>Timer 2</b> <b>Interrupt Priority Timer 2 &lt; Interrupt Priority Timer 3</b> PL(Timer 2) > PL(Timer 3)				
Interrupt-Frequenz Timer 3	Minimum Latenz <small>Anzahl ticks</small>	Maximum Latenz <small>Anzahl ticks</small>	Durchschnitt Latenz <small>Anzahl ticks</small>	Durchschnitt T <sub>ISR</sub> <small>Anzahl ticks</small>
Timer 3 off Siehe 4.1b)	21	32	21	86
10 kHz	17	148	22	88
100 kHz	17	149	31	99
1 MHz	-	-	-	-

## 4.5 Interpretation

Interpretieren Sie Ihre Messergebnisse (Aufgabe 4.4 im Vergleich zu Aufgabe 4.3c) in Bezug auf Latenz und Dauer der ISR. Was stellen Sie fest? Erklären Sie die Gründe. Was passiert bei der letzten Messung?

Fall 1: Eine erhöhte Priorität des Timers 2 verringert die minimale und maximale Latenzzeit, ohne die TISR zu und durchschnittliche Latenzzeit beeinflussen.

Fall 2: Eine erhöhte Priorität des Timers 3 erhöht die TISR, maximale Latenzzeit und die durchschnittliche Latenzzeit. Bei einer Frequenz von 1 MHz kann das Programm blockieren, da die ISR des Timers 3 die ISR des Timers 2 unterbrechen kann. Dies wirkt sich auf den Durchschnitt aus und führt zu längeren Latenzzeiten.

Bei 1 MHz bleibt das Programm kontinuierlich in der Bearbeitung von ISRs des Timers 3, während die ISR des Timers 2 keine Chance zur Ausführung hat und das Programm blockiert, da es auf eine bestimmte Anzahl von ISRs des Timers 2 wartet.

**Hinweis:** Die Resultate in den Tabellen können variieren. Sie sind als Anhaltspunkte zu verstehen.

## 5 Bewertung

Die lauffähigen Programme müssen präsentiert werden. Die einzelnen Studierenden müssen die Lösungen und den Quellcode verstanden haben und erklären können.

Bewertungskriterien	Gewichtung
Interruptroutine zur Messung der Latenz	1/4
Ausgabe der Werte auf dem Display	1/4
Messung der Latenz unter Last	1/4
Interpretation der Resultate	1/4